REMARKS

Introduction

Claims 1-3, 5-9, and 13-15 were pending. Claims 1 and 7 are independent. Claims 1 and 7 have been amended. Claims 2, 3, 5, and 6 have been cancelled. Claims 17-29 have been added. Applicants submit this amendment to put the application in condition for allowance or in better condition for appeal.

CLAIM OBJECTIONS

In the Office Action, the Examiner objected to claims 5 and 6 because of certain informalities noted by the Examiner. By this Amendment, claims 5 and 6 have been cancelled In the Office Action, the Examiner objected to claim 2 for failing to further limit claim 1. By this Amendment, the claim has been cancelled Accordingly, withdrawal of the objections to claims 2, 5, and 6 is requested.

Rejections under 35 U.S.C. § 112

Claims 1-3, 5-9, and 13-15 stand rejected under 35 U.S.C. § 112, second paragraph as allegedly being indefinite for referring to the trademark "Java Bean". By this Amendment, the pending claims have been amended to address the 35 U.S.C. § 112 rejections. More specifically, claims 1 and 7, and new claims 17-29 now refer to "queue bean, "message bean" and a "queue manager bean". From the context of the claims, it is clear what tasks the various beans perform. They are also referred to in the specification as the same. Ultimately, the various bean types derive from Java Beans, which are a construct in Java, as is known in the art. The definition of a Java Bean can be found in the Sun Microsystems Java and Java Bean specifications, which define a Java Bean as a "reusable software component that can be

manipulated visually in a builder tool." They are also defined, for example, on the Wikipedia web site as classes written an Java which encapsulate many objects into a single object (the bean), so that the bean can be passed around rather than the individual objects. Claims 2-6 have been cancelled. Accordingly, withdrawal of the rejections to claim 1, 7-9, and 13-15 under 35 U.S.C. § 112, is requested.

Rejections under 35 U.S.C. § 103(a)

Claims 1, 3-7, and 10-12 stand rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,046,742 (Chari) in view of "MIB for FIFO, Priority, Custom, and Fair Queueing," (Baker), in further view of NNRD435152 "Bloodhound Server Monitor Package" (IBM) and in further view of U.S. Patent Application Publication No. US 2004/0024579 (Nusbickel et al.).

Chari describes a method for organizing and displaying management information regarding the hardware and software components in a computer network. The software modules of Chari employ SNMP protocol with access to a management information base (MIB) to organize the data into major component groups and comprise a plurality of operational parameters about different components in a computer network. The operational parameters are organized into a plurality of hierarchical levels. The method further comprises a plurality of forms which enable the modification of one or more of the operational parameters. Each of the forms correspond to one of the hierarchical levels.

According to the Examiner, Chari discloses sending a query requesting queue status to an application server, which is the SNMP Manager. An MIB manager module calls the SNMP manager module to get the queue data (MIB data) from the network. The queue data (MIB data) is delivered to the user on a display in a tree-like structure (FIG. 16). It is a plurality

of software modules which employ the SNMP protocol which receive the queue information and render the tree structure from MIB data, the MIB data being capable of storing queue status data in a hierarchical collection of variables. The Examiner contends that Column 9, lines 34-36 of Chari inherently discloses that the data collected is sorted into categories.

In contrast to the method described by Chari, amended claims 1 and 7 of the present application recite, *inter alia*, a method and an apparatus configured to execute steps for sending a query regarding status of one or more queues and including a queue manager name to a tree renderer located on an application server which includes a queue bean; sending a message from the queue bean to retrieve a list of queues corresponding to the named queue manager to one of the plurality of message servers on multiple platforms; receiving and storing the list of queues from the one of a plurality of message servers at the queue bean; providing the list of queues to the tree renderer by the queue bean; processing the list of queues into a tree structure by the tree renderer; and delivering the-tree structure to a user in a web browser on a display.

Nowhere in Chari is it described or taught any particular method steps for obtaining the list of queues. There is no mention of selecting a queue manager, sending a message for querying queue status to an object like the tree renderer which itself performs the task of retrieving the list of queues from a queue bean and then renders the list into a tree structure. There is no mention of a queue bean which selects the queue manager responsible for managing a list of queues and then storing the list in the queue bean. The steps of how the queue data is obtained and sorted into a tree structure is specifically spelled out in amended claim 1. Chari only discloses the use of an MIB. There is no description or teaching in Chari of the steps taken to put the queue data into a tree structure in an MIB. Chari discloses using SNMP protocol executed by SNMP compatible modules, particularly the MIB Manager Module 402, and MIB Section Module 404, and MIB Variable module 406 to retrieve the queue data. As

cited by the Examiner, the SNMP Window module 416 is responsible for deriving a tree structure from retrieved MIB data. This SNMP Window Module 416 does not employ any type of (java) beans for storing the list of queues. A java bean is a class which encapsulates several classes for performing a plurality of tasks which are passed around as a single entity or object. The modules of Chari do not disclose classes that are encapsulated and passed along as a single class, only the use of a MIB under SNMP protocol. Accordingly, applicant submits that Chari does not describe or teach the invention recited by amended claims 1 and 7 of the present application.

Baker fails to correct the deficiencies of Chari. Baker merely describes SNMP code for monitoring queue status. There is no mention of selecting a queue manager, nor sending a message for querying queue status to an object like the tree renderer which itself performs the task of retrieving the list of queues from a queue bean and then renders the list into a tree structure. There is no mention of a queue bean which selects the queue manager responsible for managing a list of queues and then storing the list in the queue bean. There is no mention of how queue data is obtained from a queue manager and sorted into a hierarchical structure. Merely mentioning that the MIB is inherently sorted does not explain how the data gets sorted for storage in the MIB.

IBM fails to correct the deficiencies of Chari and Baker. As cited by the Examiner, IBM describes using a web browser to access a network monitoring program. There is no mention of selecting a queue manager, nor sending a message for querying queue status to an object like the tree renderer which itself performs the task of retrieving the list of queues from a queue bean and then renders the list into a tree structure. There is no mention of a queue bean which selects the queue manager responsible for managing a list of queues and then storing the

list in the queue bean. There is no mention of how queue data is obtained from a queue manager and sorted into a hierarchical structure.

Nusbickel et al. fails to correct the deficiencies of Chari, Baker, and IBM. Nusbickel et al. discloses a method of adding an SNMP interface to an existing resource management extension-enabled management agent such as one implemented in Java using Java Beans. There is no mention of selecting a queue manager, nor sending a message for querying queue status to an object like the tree renderer which itself performs the task of retrieving the list of queues from a queue bean and then renders the list into a tree structure. There is no mention of a queue bean which selects the queue manager responsible for managing a list of queues and then storing the list in the queue bean. There is no mention of how queue data is obtained from a queue manager and sorted into a hierarchical structure.

As such, withdrawal of the rejection of claims 1 and 7 under 35 U.S.C. 103(a) based on Chari in view of Baker in further view of IBM and in further view of Nusbickel et al. is requested.

Each of new claims 17-23 ultimately depend from claim 1; each of pending claims 8, 9, 13-15 and new claims 24-29 ultimately depend from claim 7. Dependent claims 8, 9, 13-15, and 17-29 are deemed to be patentable over Chari in view of Baker in further view of IBM and in further view of Nusbickel et al., for at least the reasons described above with respect to the patentability of claims 1 and 7.

Thus, applicant submits that each of the claims of the present application are patentable over each of the references of record, either taken alone, or in any proposed hypothetical combination. Accordingly, withdrawal of the rejections to the claims is respectfully requested.

Conclusion

In view of the above remarks, reconsideration and allowance of the present application is respectfully requested. If any fees are deemed necessary for this Amendment to be entered and considered by the Examiner, then the Commissioner is authorized to charge such fee to Deposit Account No. 50-1358. Applicant's undersigned patent agent may be reached by telephone at (973) 597-2500. All correspondence should continue to be directed to our address listed below.

Respectfully submitted,

Date: 7/3/07

Raymond G. Cappo
Patent Agent for Applicant
Registration No. 53,836

DOCKET ADMINISTRATOR
LOWENSTEIN SANDLER PC
65 Livingston Avenue
Roseland, NJ 07068